

INSIDE DOS

Tips & techniques for MS-DOS & PC-DOS Versions 5 & 6

VERSION
5.0 & 6.x

Is there a new operating system in your future?

By Van Wolverton

You care enough about DOS to read *Inside DOS*, so you're probably aware that MS-DOS has some challengers to its position as the leading operating system for the PC world. Some of those challengers are alternative Microsoft products, but some interesting (and significant) non-Microsoft products are also vying for the chance to operate your system. The turmoil we've seen in the operating systems market in recent years is just the beginning of this battle.

Since Version 1.0 first ran on an IBM PC nearly 13 years ago, DOS has been gussied up with lots of nifty new capabilities. In fact, DOS is hardly recognizable as the same program—unless your command prompt remains an unvarnished C>. However, DOS possesses some inherent limitations that make it difficult, if not impossible, for software to take full advantage of today's hardware. The availability of computers based on the PowerPC microprocessor—a joint venture of Apple, IBM, and Motorola intended to combat the market dominance of Microsoft and Intel—will only intensify the demand for operating systems free from these limitations.

So how does all this affect you? You have a lot of time and money invested in your software. More important, you have hundreds, if not thousands, of data files that work on your current system. You'll need a powerful incentive to abandon your programs, learn new programs, and somehow convert those data files. How much longer can you continue to use your comfortably familiar programs on your trusty PC?

Fortunately, with few exceptions, you can stick with your tried-and-true programs—including the venerable DOS—just as long as you like without significantly sacrificing speed or convenience. Circumstances may eventually motivate you to switch operating systems, but the market won't abandon DOS applications for quite a few years.

The limitations of DOS

DOS is the most frequently and vehemently criticized piece of PC software ever developed. Much of the criticism about its shortcomings is gratuitous,

made without regard for what the great majority of the market needs today or what the market required (or seemed to require) when the first version was written. With all its failings, DOS has been one of the two major factors in the remarkable success of the personal computer market in the past 13 years. (The other factor was IBM's decision to make the PC an open system.)

All that said, there's no escaping DOS's shortcomings. The 640 Kb memory limit is the most infamous of these deficiencies, but the file structure and task management—the way it controls other programs—are equally limiting. However, as hardware gets more powerful and software developers demand better access to hardware capabilities, DOS's fundamental limitations will eventually force its demise.

Prompted in part by these limitations, six alternatives to MS-DOS are available today: Windows 3.1 and Windows NT from Microsoft; PC-DOS 6.1, OS/2 Version 2.1, and OS/2 for Windows from IBM; and Novell DOS 7.0 from Novell. Due out this year or next are Windows 4.0 (the much-hyped "Chicago"), a new version of Windows NT, and OS/2 Version 3.0. They each have something to offer that MS-DOS doesn't; they each exact a price; and they each offer some additional risk.

IN THIS ISSUE

- Is there a new operating system in your future? 1
- Tracking the results of changes to your system configuration 3
- Creating and modifying text files in DOS 6
- Entering ASCII and control characters into a text file 9
- Disabling DOS 6's [F5] and [F8] keys at startup 12

Windows was the first alternative...

Microsoft offered the first alternative to DOS with Windows. It's really an addition to DOS, not a replacement. Although the Windows marketing team at Microsoft intended that users install Windows and never see the DOS prompt again, this hasn't been the case. Most PC users use Windows to run programs that require it but still use plain old DOS much of the time. If you use Windows, you probably fit that pattern. (For example, I use Windows mostly for PageMaker, Corel Draw, and Excel; the rest of the time I use DOS.)

Windows requires a faster system with more memory than DOS, and many programs still run more slowly than an equivalent program would run under DOS. Although Windows has been available for eight years, it still fails much more often than programs should (case in point: your system locks up for no apparent reason and your only clue is the dreaded *General Protection Failure* message). When was the last time DOS failed in your machine?

Windows NT is like Windows, but on a larger scale. It requires an even faster system with still more memory (an unbelievable 20 Mb). NT is used primarily by very large network installations and software developers. It isn't an operating system alternative for the majority of users.

IBM offers a few more alternatives...

IBM and Microsoft parted company starting with Version 6 of DOS. Microsoft offered MS-DOS 6.0 and IBM, playing a little one-upmanship, came out with PC-DOS 6.1 a few months later. Then Microsoft came out with a damage-control release called Version 6.2 to quell horror stories about DoubleSpace, the file-compression utility included with MS-DOS 6.0. PC-DOS 6.1 is nearly identical to MS-DOS 6.0 and 6.2 because, at the time it was released, IBM still had the right to use Microsoft's actual code. That agreement has expired now, so IBM's next version may not be quite as similar.

OS/2 was also an IBM-Microsoft joint venture, but differences in market plans for that operating system played a significant role in the deterioration of the relationship between those two computer leviathans. IBM is now using OS/2 to attack Microsoft's share of the operating system market. Unlike Windows, OS/2 replaces DOS and offers what both DOS and Windows lack: efficient, painless management of huge amounts of memory, and true multitasking that lets you run just about as many programs as you like. As a bonus, OS/2 lets you run DOS, Windows, and OS/2 programs simultaneously.

OS/2 needs more power and memory than DOS to run well, but no more than Windows requires. OS/2 certainly doesn't have the gargantuan appetite for memory and processing power that Windows NT has. It's defi-

INSIDE DOS

Inside DOS (ISSN 1049-5320) is published monthly by The Cobb Group.

Prices: Domestic: \$49/yr (\$6.00 each)
Outside US: \$69/yr (\$8.50 each)

Phone: Toll free: (800) 223-8720
Local: (502) 491-1900
Customer Relations Fax: (502) 491-8050
Editorial Department Fax: (502) 491-4200

Address: You may address tips, special requests, and other correspondence to
The Editor, *Inside DOS*
9420 Bunsen Parkway, Suite 300
Louisville, KY 40220

For subscriptions, fulfillment questions, and requests for bulk orders, address your letters to

Customer Relations
9420 Bunsen Parkway, Suite 300
Louisville, KY 40220

Copyright: Copyright © 1994, The Cobb Group. All rights reserved. *Inside DOS* is an independently produced publication of The Cobb Group. The Cobb Group reserves the right, with respect to submissions, to revise, republish, and authorize its readers to use the information submitted for both personal and commercial use.

The Cobb Group, its logo, and the Satisfaction Guaranteed statement and seal are registered trademarks of Ziff Communications Company. *Inside DOS* is a trademark of Ziff Communications Company. Microsoft and MS-DOS are registered trademarks and Microsoft Windows is a trademark of Microsoft Corporation. PC-DOS is a trademark of IBM Corporation.

Postmaster: Second class postage is pending in Louisville, KY. Send address changes to

Inside DOS
P.O. Box 35160
Louisville, KY 40232

Authorized Canada Post International Publications Mail (Canadian Distribution) Sales Agreement #XXXXXX CANADA GST #123669673. Send returns to Canadian Direct Mailing Sys. Ltd., 920 Mercer Street, Windsor, Ontario, N9A 7C2. Printed in the USA.

Staff: Editor-in-Chief: Janice Walter
Contributing Editor: Van Wolverton
Editors: Mary Jacobson
Linda Recktenwald
Cecilia Crosby-Lampkin
Tessa Gavron
Production Artists: Kate Stites
Karen Collins
Julie Jefferson
Managing Editor: Mark Kimbell
Circulation Manager: Marjorie Glassman
Editorial Director: Jeff Yocom
Publishers: Mark Crane
Jon Pyles

Advertising: For information about advertising in Cobb Group journals, contact Tracee Bell Troutt at (800) 223-8720, ext. 430.

Back Issues: To order back issues, call Customer Relations at (800) 223-8720. Back issues cost \$6.00 each, \$8.50 outside the US. You can pay with MasterCard, VISA, Discover, or American Express, or we can bill you.

Advisory Board: Earl Berry Jr.
Tina Covington
Marvin D. Livingood

nately not easy to install, and getting it to run properly requires an inordinate amount of tinkering that makes working with CONFIG.SYS and AUTOEXEC.BAT seem like kindergarten exercises. Another bonus: OS/2 is much more stable than Windows. All things considered, however, it isn't ready yet to supplant DOS on the tens of millions of PCs currently in homes and small businesses.

... and Novell gives you another choice

Novell DOS is especially tempting to those who have complained for years about the shortcomings of Microsoft's DOS, but it doesn't offer a compelling reason for most users to switch. Novell DOS 7.0 is a near-clone of MS-DOS that offers more capability—for example, you can run a local area network with no additional software, and it lets you run more than one program at once. But this version of DOS requires more tinkering than MS-DOS or PC-DOS and isn't totally compatible with either.

Most systems sold today come with Windows already installed, and most of the new software is written for Windows, not for DOS. Many software developers don't write programs for DOS anymore, and others have announced that they won't be updating their existing DOS programs. But there are still plenty of DOS programs available for nearly every type of application. Not everyone wants or needs to pay for a more powerful system—especially one that's slower—just to use a graphics-based operating system.

And be glad you aren't a Macintosh user. Not only have they faced similar hassles as new (and buggy) versions of their Mac operating system became available, but their operating system future is filled with new choices—products with code names like Mozart,

Copland, Gershwin, and KN—and they're facing the unsettling fact that their systems are becoming more and more PC-compatible.

Make haste slowly

If you work for a large company or if you're a freelancer with clients who use Windows or OS/2, you probably have no choice: For work, at least, it's going to be a graphical user interface world for you. But if you're in charge of your own destiny, don't let yourself be pressured into intemperate action. Even in this lightning-fast world of desktop computers, things change more slowly than events would seem to dictate.

Most of us are paying bills that are printed by programs written 30 years ago in COBOL. Some of these programs are running on modern computers that emulate the behavior of an IBM 1401, a mainframe computer IBM stopped manufacturing in the early 1960s that has less power than a 33 MHz 386. It's a woefully inefficient system, but the bill-printing program works, and it's easier and cheaper to find ways to continue using the same program than to write a new version of the program.

So don't start looking for a replacement operating system just yet. Even in the year 2000 you'll still be able to use DOS and get all kinds of work done with your computer. You may not have seen a new version of your programs for years, and you may decide that it really is time to run all that jazzy new software. But most important, you'll be pulled into the brave new world, not pushed.

Contributing editor Van Wolverton has worked for IBM and Intel and is the author of the books Running MS-DOS 5 and Supercharging MS-DOS. ♦

WORKING SMARTER

VERSION
5.0 & 6.x

Tracking the results of changes to your system configuration

As most of us learn more about DOS, we tweak our CONFIG.SYS and AUTOEXEC.BAT files to try to gain a little memory here and a little speed there. If you're in this group, you've undoubtedly discovered you can use the MEM command and its switches to see how the configuration changes you make affect your computer's memory.

Getting the optimal system configuration is often a matter of trial and error. You make a few changes to CONFIG.SYS and/or AUTOEXEC.BAT, then reboot and run MEM to look at your new memory setup. Then

you make a few more changes, reboot, and check your memory again. After a few rounds, it's hard to remember which changes had what effect. If this situation sounds familiar, we can help.

In this article, we won't suggest ways to configure your system, but we'll show you a batch file that will give you the tools to help you with those decisions. The batch file will compile and print the relevant configuration information so you can compare how your various changes affect your system's memory usage. (As always, we recommend that you

save copies of AUTOEXEC.BAT and CONFIG.SYS under different names before modifying them.)

Creating and printing a configuration text file

Creating a text file containing your configuration information is an easy process but consists of several steps. During the process, you add headers and blank lines to make the text file readable, you type the contents of CONFIG.SYS and AUTOEXEC.BAT to the file, and then you direct the output of the MEM command to the file.

You build and print the text file by following these steps:

1. Create the file and add a header for the CONFIG.SYS file by using the command

```
echo CONFIG.SYS: > filename
```

By using the > redirection operator, you create the text file as you add information to it.
2. Send a blank line to separate the header from the contents of the CONFIG.SYS file with the command

```
echo. >> filename
```

where *echo.* creates a blank line. By using the >> operator, you append the blank line to the end of the file you created in Step 1.
3. Place the contents of the CONFIG.SYS file at the end of the file using the command

```
type config.sys >> filename
```
4. Add two more blank lines by using the command in Step 2 twice.
5. Place a header for the AUTOEXEC.BAT file by using the command

```
echo AUTOEXEC.BAT: >> filename
```
6. Echo another blank line to the file.
7. Append the contents of AUTOEXEC.BAT to the file using the TYPE command as you did in Step 3.
8. Echo two more blank lines to the file.
9. Add a header for the output of the MEM command by using the command

```
echo MEMORY REPORT: >> filename
```
10. Append the contents of the MEM command by using the command

```
mem /switch >> filename
```

where */switch* is the switch or switches that will give you the specific type of information you want. (You can type *MEM /?* to see the switches available for the MEM command.) Because the

output of the MEM command is text, DOS lets you redirect the output straight into a text file without having to use the TYPE command.

11. Print the text file by using the command

```
print filename
```

Now that you know the steps involved in manually creating and printing a configuration text file, you can see why you'll want to place the commands in a batch file. A batch file is the perfect solution for this kind of repetitive task, so let's create one.

Creating MEMORY.BAT

MEMORY.BAT automates the steps involved in creating and printing a report that contains your CONFIG.SYS file, AUTOEXEC.BAT file, and the output of the MEM command. The batch file lets you enter a MEM command switch when you invoke it in order to specify the type of memory report you want.

To create the batch file, open the DOS Editor or another ASCII-compatible word processor and type the batch file as it appears in Figure A. If you use the DOS Editor, you can take advantage of its Copy and Paste features to avoid typing the repeated ECHO commands. Copy the first *echo. >> z.zzz* command to the Clipboard by highlighting it and pressing [Ctrl][Insert]. Then copy the command from the Clipboard to your batch file by moving the cursor to a new line and pressing [Shift][Insert]. Save the batch file under the name MEMORY.BAT in a directory listed in your PATH statement and exit to DOS.

Our batch file follows the same steps we discussed earlier, but with a few additions. The batch file begins with the typical @ECHO OFF and REM commands. Then it displays the messages in the :REMINDERS section, which reminds you to turn your printer on and asks if you remembered to reboot before running MEMORY.BAT.

Next, the :COMPILING section creates the configuration information file under the dummy filename Z.ZZZ, following Steps 1 through 9 as we discussed earlier. The last command in this section

```
mem %1 >> z.zzz
```

uses the %1 parameter to hold the switch you include when you run the batch file. This command compiles the specific memory report you want and appends it to the configuration text file. (You may want to specify a MEM switch to use every time you run the batch file. If so, just replace the %1 parameter with one of MEM's switches.)

The :PRINTING section prints the text file and then deletes it. We initialized the printer by specifying *prn* as the printer name. If you use a different printer port, just add the appropriate name to the PRINT

statement. If you initialize your printer in your AUTOEXEC.BAT file, you can remove the /D:PRN switch from MEMORY.BAT.

Figure A

```
@echo off
rem MEMORY.BAT creates a text file that holds your
rem configuration files and details your current
rem memory usage.

:REMINDERS
echo Make sure your printer is on and is online.
echo.
echo Did you remember to reboot your computer?
echo If not, reboot and run MEMORY again.

:COMPILING
echo CONFIG.SYS file: > z.zzz
echo. >> z.zzz
type C:\CONFIG.SYS >> z.zzz
echo. >> z.zzz
echo. >> z.zzz
echo AUTOEXEC.BAT file: >> z.zzz
echo. >> z.zzz
type C:\AUTOEXEC.BAT >> z.zzz
echo. >> z.zzz
echo. >> z.zzz
echo MEMORY REPORT: >> z.zzz
mem %1 >> z.zzz

:PRINTING
print /d:prn z.zzz
del z.zzz

:END
```

MEMORY.BAT creates a text file containing your configuration files and a report of the current memory usage.

Using MEMORY.BAT

MEMORY.BAT is easy to use. First you save your new configuration, reboot, and make sure your printer is turned on and is online. Then you simply enter the command's name followed, optionally, by the switch that will give you the information you need:

`memory /switch`

(If you specify a switch from inside the batch file, just enter the batch file's name.)

The most important part of using MEMORY.BAT is remembering to reboot your computer after you modify CONFIG.SYS and/or AUTOEXEC.BAT but before you run the batch file, since your new configuration doesn't take effect until you reboot your computer. That way, your memory report will reflect your new configuration.

Once you have a printout, you can review your configuration files and how they affect memory usage. At this point, you can make more changes to CONFIG.SYS and AUTOEXEC.BAT, reboot, and run the batch file again. Soon you'll find just the right configuration for your system. Now let's use MEMORY.BAT to print a report.

An example

Let's say you want to make some changes to your DOS 6.2 system configuration. Since you haven't changed your CONFIG.SYS or AUTOEXEC.BAT file, you should run MEMORY.BAT now so you'll have a record of your original memory usage. First, press [Ctrl][Alt][Delete] to reboot your computer in order to flush any memory-resident programs from your system. Then, enter the name of the batch file followed by the /CLASSIFY, or /C, switch

```
C:\>memory /c
```

Including the /C switch instructs the batch file to create and print your CONFIG.SYS file, your AUTOEXEC.BAT file, and a report that lists the size and location of programs in conventional memory and upper memory as well as a summary of each type of memory currently in use. Figure B shows the report MEMORY.BAT creates.

Figure B

```
CONFIG.SYS file:
DEVICE=C:\DOS\HIMEM.SYS /v
DEVICE=C:\DOS\EMM386.EXE NOEMS
BUFFERS=20,0
FILES=40
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\ANSI.SYS
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /p
STACKS=9,256
DEVICEHIGH=C:\DOS\DBLSPACE.SYS /MOVE

AUTOEXEC.BAT file:
@echo off
LH C:\dos\smartdrv.exe
prompt $p$g
set mouse=c:\mouse
LH C:\mouse\mouse.exe
PATH C:\DOS;c:\c:\windows;c:\dos5;c:\batch;c:\gu\collage;c:\mouse;c:\buttons
set temp=c:\dos
LH C:\dos\doskey
doskey whereis = dir /s/p $1

MEMORY REPORT:
Modules using memory below 1 MB:

Name          Total      =  Conventional  +  Upper Memory
-----
MSDOS          64,621  (63K)    64,621  (63K)    0  (0K)
HIMEM           3,792  (4K)     3,792  (4K)    0  (0K)
EMM386          3,120  (3K)     3,120  (3K)    0  (0K)
SETVER           592  (1K)       592  (1K)    0  (0K)
ANSI             4,208  (4K)     4,208  (4K)    0  (0K)
DBLSPACE        53,440  (52K)    53,440  (52K)    0  (0K)
COMMAND         5,072  (5K)     5,072  (5K)    0  (0K)
SMARTDRV        27,488  (27K)    27,488  (27K)    0  (0K)
MOUSE            272  (0K)       272  (0K)    0  (0K)
DOSKEY          4,144  (4K)     4,144  (4K)    0  (0K)
Free            488,496 (477K)  488,496 (477K)    0  (0K)

Memory Summary:

Type of Memory  Total  =  Used  +  Free
-----
Conventional    655,360  166,864  488,496
Upper           0         0         0
Reserved        0         0         0
Extended (XMS)  3,538,944  1,429,504  2,109,440
Total memory    4,194,304  1,596,368  2,597,936
Total under 1 MB  655,360  166,864  488,496

Largest executable program size  488,192 (477K)
Largest free upper memory block  0 (0K)
The high memory area is available.
```

MEMORY.BAT created this report when we ran it with the /C switch in DOS 6.2.

Now, you want to change your CONFIG.SYS file by adding a command that will move DOS to the upper memory area. Using the DOS Editor, open the CONFIG.SYS file and add the line shown in red in Figure C. Save the new CONFIG.SYS file and exit the Editor.

Figure C

```
DEVICE=C:\DOS\HIMEM.SYS /v
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
BUFFERS=20,0
FILES=40
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\ANSI.SYS
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /p
STACKS=9,256
DEVICEHIGH=C:\DOS\DBLSPACE.SYS /MOVE
```

Modify your CONFIG.SYS file by adding the line shown in red.

To see how this simple change affects your conventional memory and upper memory, reboot your computer. Then run the batch file again using the same command:

```
C:\>memory /c
```

This time, MEMORY.BAT produces the report shown in Figure D. As you can see by comparing the Memory Summary sections of the two reports, adding the line that moves DOS to high memory recaptures a whopping 132 Kb of conventional memory space! Also, this command lets DOS obey the CONFIG.SYS DEVICEHIGH command and the AUTOEXEC.BAT LOADHIGH command, which load various device drivers and programs into the upper memory area as well.

At this point, you can decide whether you made the right changes to your configuration and whether you want to continue tweaking your system files.

Figure D

CONFIG.SYS file:

```
DEVICE=C:\DOS\HIMEM.SYS /v
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
BUFFERS=20,0
FILES=40
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\ANSI.SYS
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /p
STACKS=9,256
DEVICEHIGH=C:\DOS\DBLSPACE.SYS /MOVE
```

AUTOEXEC.BAT file:

```
@echo off
lh c:\dos\smartdrv.exe
prompt $p$g
set mouse=c:\mouse
lh c:\mouse\mouse.exe
PATH C:\DOS;c:\c:\windows;c:\dos5;c:\batch;c:\gu\collage;c:\mouse;c:\buttons
set temp=c:\dos
lh c:\dos\doskey
doskey whereis = dir /s/p $1
```

MEMORY REPORT:

Modules using memory below 1 MB:

Name	Total	=	Conventional	+	Upper Memory
MSDOS	16,813 (16K)		16,813 (16K)		0 (0K)
HIMEM	1,168 (1K)		1,168 (1K)		0 (0K)
EMM386	3,120 (3K)		3,120 (3K)		0 (0K)
SETVER	592 (1K)		592 (1K)		0 (0K)
ANSI	4,208 (4K)		4,208 (4K)		0 (0K)
COMMAND	5,072 (5K)		5,072 (5K)		0 (0K)
MOUSE	24,368 (24K)		272 (0K)		24,096 (24K)
DBLSPACE	39,472 (39K)		0 (0K)		39,472 (39K)
SMARTDRV	27,488 (27K)		0 (0K)		27,488 (27K)
DOSKEY	4,144 (4K)		0 (0K)		4,144 (4K)
Free	687,344 (671K)		623,968 (609K)		63,376 (62K)

Memory Summary:

Type of Memory	Total	=	Used	+	Free
Conventional	655,360		31,392		623,968
Upper	158,576		95,200		63,376
Reserved	0		0		0
Extended (XMS)	3,380,368		1,270,928		2,109,440
Total memory	4,194,304		1,397,520		2,796,784
Total under 1 MB	813,936		126,592		687,344

Largest executable program size 623,872 (609K)
Largest free upper memory block 52,432 (51K)
MS-DOS is resident in the high memory area.

Compare this report with the original to see how the changes you made to CONFIG.SYS affect your memory usage.

Conclusion

Finding just the right configuration for your system is a difficult task. Sometimes the hardest part is remembering what changes you’ve made to your system files and how those changes affected your computer’s memory usage. In this article, we showed you a batch file you can use to compile a report that includes your CONFIG.SYS file, AUTOEXEC.BAT file, and a summary of how your computer’s memory is distributed. ♦

Creating and modifying text files in DOS

When you want to create a text file in DOS, you might automatically open your word processing program, type the text, save it, and exit back to DOS. However, that’s a lot of steps if you’re creating a short text file that doesn’t in-

clude fonts, graphics, or special formatting. DOS offers several ways to create text files and to add text to files that already exist. In this article, we’ll briefly describe how you can create and modify text files in DOS.

Creating a one-line file using ECHO and the > redirection symbol

You can use the ECHO command from the DOS prompt to place a single line of text into a file. For example, using the ECHO command to create a text file is very handy when you need a dummy file for testing batch files and DOS commands. Simply use a command in the form

```
echo text > filespec
```

where *text* is the text string you want to save and *filespec* is the path and name of the file in which you want to save the text string. You can create a file named HGWELLS.TXT by entering

```
C:\>echo No passion in the world is equal to the  
passion to alter someone else's draft.----H.G.  
Wells> hgwells.txt
```

As with all commands you can enter from the DOS prompt, the length of the entire command must be 127 characters or less. If you discover a typing error, you can edit it before you press [Enter] to save the file. Just press the ← key until you've deleted the typo. Then retype the rest of the line. Or, if Doskey is loaded, press [Insert] and use the arrow keys to move the cursor to the error. Then delete the typo, type the correction, and press [End] to return the cursor to the end of the line.

ECHO does have its limits, though. You can't use ECHO to place redirection symbols (>, <, >>) or the pipe operator (|) in a text file because ECHO interprets these characters as commands, not as text. Also, you can't echo certain control characters to a text file because DOS interprets them as commands and tries to execute them immediately. (We discuss control characters in the article "Entering ASCII and Control Characters into a Text File" on page 9.) And, if you accidentally save a typing error, you'll have to start over or use the Editor (in DOS 5 or 6.x) or Edlin (available through DOS 5) to fix it.

Creating simple, multiline text files

If you want to create a short text file, consider using the COPY CON command. COPY CON is especially useful for creating batch files that are only a few lines long. The form of the command is

```
copy con filespec
```

where *filespec* is the path name and filename of the file you want to create. When you enter the command, the cursor will drop to a blank line below the prompt. At this point, you should type the first line of text and press [Enter]. The cursor will drop to the next line, and you can enter another line of text.

Watch your typing! You can't edit a line once you press [Enter]—the line becomes part of the file. If you find a mistake on a previous line, you'll have to quit and

start over or edit the offending line in the Editor or Edlin.

Once you finish, press either [F6] or [Ctrl]Z to place the ^Z character at the end of the file. Then press [Enter], and DOS shows the message *1 file(s) copied* and returns you to the prompt. For example, you can create a text file containing the following bit of literary nationalism with the COPY CON command:

```
C:\> copy con cottarel.txt  
In America only the successful writer is important,<[Enter]>  
in France all writers are important, in England no <[Enter]>  
writer is important, in Australia you have to <[Enter]>  
explain what a writer is. <[F6]><[Enter]>
```

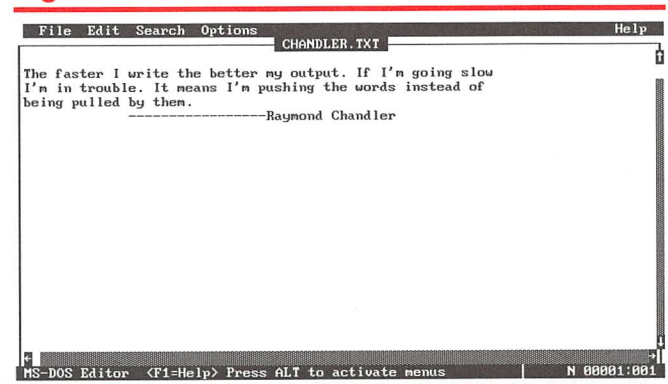
Using the DOS Editor to create and edit files

When you need to create a large text file or modify an existing file, you can't beat the MS-DOS Editor for getting the job done. You start the Editor and bypass the startup screen by using the command

```
edit filespec
```

where *filespec* is the path and filename of the file you want to create or edit. Unlike other text editors, the Editor lets you view and edit the entire file at once. Figure A shows the CHANDLER.TXT file after we created it in the Editor, but before we saved it and exited.

Figure A



The Editor lets you view and edit the entire file.

The Editor offers several commands and options from menus. With the Editor, you can:

- print the file
- globally search and replace words and characters
- cut or copy text and paste it elsewhere in the file
- search for a specific word or character
- save the file under a different name

You can even change the Editor's display colors or press [F1] to get help on using the Editor. When you exit the Editor by pressing [Alt]FX, it even alerts you if you haven't saved your file and asks whether you now want to save it.

Using Edlin in DOS 5

Edlin is the granddaddy of all DOS text editors and was, before DOS 5, the only DOS editor. Many users swear by Edlin; most swear off Edlin after trying to use it once. However, Edlin is useful for making minor changes to text files when you don't want to take the time to open the Editor. To open a new or old file, enter the command

```
edlin filespec
```

If you're creating a file, you're greeted with the message *New file* followed by a line that begins with an asterisk. At this point, enter *I* (for insert) and begin typing at the numbered line Edlin presents. Press [Enter] at the end of line 1 and begin typing at line 2. Once you've finished inserting text, press [Ctrl]C to end insert mode.

If you're editing an existing file, Edlin presents the message *End of input file* followed by the asterisk prompt. At this point, enter *L* (for list) and Edlin will display the file with each line preceded by a number. To edit a particular line, enter that line's number and begin typing. When you press [Enter] at the end of the line, Edlin returns you to the asterisk prompt.

Whether you're editing or inserting, you can exit Edlin in one of two ways: You can save the changes by entering *E* or you can stop the session without saving the changes by entering *Q*. For example, you can create a new file by entering text as shown in color below:

```
C:\>edlin twain.txt
New file
*i
1:*The man who does not read good books has <[Enter]>
2:*no advantage over the man who can't <[Enter]>
3:*read them. <[Enter]>
4:* ----- Mark Twain <[Enter]>
5:*<[Ctrl]C>
*e
```

(The [Ctrl]C keypress will appear as ^C onscreen.) Your DOS 5 user's guide gives a full description of the many options and commands you can use with Edlin.

Adding text to the end of a file using ECHO and the >> append symbol

You can use the ECHO command and the append redirection symbol (>>) to *very carefully* add a line of text to the end of an existing file. For example, you can add the filename at the bottom of a file; also, you can add commands to your CONFIG.SYS or AUTOEXEC.BAT file using this method. (As always, we recommend you make no changes to these files until you save a copy of them under different names.)

To append text to the end of a file, enter the command

```
echo text >> filespec
```

For example, you can add the name of the author to the quote in COTTEREL.TXT by entering the command

```
C:\>echo ----- Geoffrey Cotterell >> catterel.txt
```

Be very deliberate when using this method, however. If you accidentally use a single > symbol, you'll wipe out the existing text, and the file will contain only the new, single line of text.

Concatenating text files

Our final method of creating text files in DOS involves joining (or *concatenating*) several text files into a single text file. To perform this task, you use the COPY command and a plus sign (+) in the form

```
copy filespec1 +filespec2 +...filespecn destination
```

where you want to join the files represented by *filespec#* into a destination file. (It doesn't matter whether you include a space on either side of the plus sign.) You can include drive letters and path names in both *filespec#* and *destination*. If you don't specify a destination file, DOS will append all the files to *filespec1*. You can join as many files as you can fit within the 127-character limit DOS imposes on commands you enter at the command line.

To join all the files we created earlier into a single LITQUOTE.TXT file, for example, you enter the command

```
C:\>copy hgwells.txt + catterel.txt + chandler.txt +
      twain.txt litquote.txt
```

When you use a TYPE command to view the contents of LITQUOTE.TXT, you'll see the text shown in Figure B.

Figure B

No passion in the world is equal to the passion to alter someone else's draft.----H.G. Wells

In America only the successful writer is important, in France all writers are important, in England no writer is important, in Australia you have to explain what a writer is.----- Geoffrey Cotterell

The faster I write the better my output. If I'm going slow I'm in trouble. It means I'm pushing the words instead of being pulled by them.

-----Raymond Chandler

The man who does not read good books has no advantage over the man who can't read them.

----- Mark Twain

You can combine several text files using the COPY + command.

Of course, you'd want to clean up the appearance of the final text file; but it's worth the extra work if it keeps you from having to manually type the text of several small files into a larger file. ♦

Entering ASCII and control characters into a text file

There are a number of reasons why many DOS users are unsure how to enter control characters and other ASCII characters into text files. First, DOS offers several ways to create text files, and each method carries its own rules for entering non-keyboard characters. Second, even when you're using a single method, DOS seems to treat certain ASCII and control characters differently than it treats others. And third, much of what you can do with ASCII characters depends on how your system is set up and what devices are attached. No wonder we get confused!

In this article, we'll discuss the difference between ASCII characters and control characters. Then we'll show you how to enter ASCII and control characters into a text file using the ECHO and COPY CON commands, the MS-DOS Editor, and DOS 5's Edlin.

The ASCII character set

Simply put, the Extended ASCII character set, as we commonly call it, consists of 256 codes. Each code represents a character. Originally, ASCII consisted of 128 codes: Characters 0 through 31 are reserved for control characters (more on that shortly); and characters 32 through 127 represent the numeric, alphabetic, and punctuation keys on a keyboard.

Currently, most systems recognize 256 codes: the original 128 and an additional 128 codes, which are commonly called *extended characters*. The additional codes consist of several alphabetical characters from European languages, scientific characters, and graphics characters.

You can find a table containing the English-language ASCII character set in your DOS user's guide. For the sake of our discussion, we'll treat ASCII characters in three distinct categories: control characters (0 through 31), standard ASCII characters (32 through 127), and extended ASCII characters (128 through 255). First, we'll show you how to enter standard and extended ASCII characters into a text file.

Entering standard and extended ASCII characters into a text file

The standard ASCII characters (32 through 127) represent the keys that are available on a standard keyboard. You enter these characters simply by typing them on the keyboard. For example, you create the character A, which is ASCII character 65, by pressing [Shift]a on your keyboard. (Alternatively, you could create the A by holding down the [Alt] key and typing 65 on the numeric keypad, but of course, you wouldn't go to the trouble.)

Most of the standard ASCII characters are self-explanatory. The exceptions are ASCII character 32,

which is the equivalent of pressing [Spacebar], and character 127, which is the equivalent of pressing [Backspace].

Entering ASCII characters 128 to 255 into a text file is also easy. Regardless of which method you use to create a text file, you enter ASCII characters 128 to 255 in the same way. From the command line, the DOS Editor, and Edlin, you create an extended ASCII character by holding down the [Alt] key and typing the character's ASCII code number on the numeric keypad (make sure NUM LOCK is on). When you do, DOS immediately displays the ASCII symbol that corresponds to the code number you typed. For example, you can use the ECHO command to enter the character ≈ into a text file, like this

```
C:\>echo <[Alt]247> > equiv.txt
```

where you hold down the [Alt] key and type 247 on the numeric keypad. Onscreen, the command appears like this:

```
C:\>echo ≈ > equiv.txt
```

Control characters

DOS reserves ASCII characters 0 through 31 for control characters. We show the ASCII control characters in Table A. Control characters provide a way for you to communicate from your keyboard to devices such as printers, your monitor, and modems. For example, the control character L sends a form feed to most printers. The printer ejects the current page when it encounters a form feed command. You can put the form feed character in a text file to force a page break in the document when you print it.

Table A: The ASCII control characters

0	11	22
1	12	23
2	13	24
3	14	25
4	15	26
5	16	27
6	17	28
7	18	29
8	19	30
9	20	31
10	21	

Most control characters appear as ASCII symbols when you type or echo them to the screen. However, you won't see the symbol when you send to a device a character it interprets as a command—you'll see an action instead. For example, the form feed character displays like this when you echo it to the screen:

```
C:\>echo ^L
␣
```

However, when you send the form feed character to a printer like this

```
C:\>echo ^L > prn
```

you don't see the ␣ symbol onscreen. Instead, the printer ejects a page.

In general, you create a control character by pressing [Ctrl] plus a letter. When you do, the character appears onscreen as a caret (^) followed by a letter or other character. For example, you create the form feed character by pressing [Ctrl]L. This character displays as ^L on the command line. The ASCII code number for a control character corresponds to that number's letter in the alphabet: ASCII character 1 is ^A, ASCII 2 is ^B, and so forth.

You can use many of the control characters as graphic characters in a text file. However, you need to remember that one of your devices might interpret the control characters you send it in a text file. For example, you might want to dress up a text file with a musical symbol ♯, which is the ASCII character for ^N. When you type the file to the screen, the symbol appears just as you intended. But if you're using an Epson-compatible printer and you print your text file, you might be surprised to see your file printing oddly. However, there's a simple explanation: Epson-compatible printers interpret ^N as a command to start expanded type.

Several control characters direct communication between the keyboard and the monitor, so you can't enter them into a text file as graphic characters. The system executes the following commands either as you try to enter them into the text file or when you type the text file:

- | | | |
|----|--------------|---|
| ^G | ASCII 7 (␣) | • creates a beep |
| ^H | ASCII 8 (␣) | • backspaces |
| ^I | ASCII 9 (␣) | • tabs over five spaces |
| ^J | ASCII 10 (␣) | • creates a line feed if Doskey isn't loaded—scrolls the DOS prompt up one row at a time, creating a blank line each time you press [Ctrl]J |
| ^M | ASCII 13 (␣) | • causes a carriage return |
| ^Z | ASCII 26 (␣) | • ends the file |
| ^[| ASCII 27 (␣) | • acts as the [Escape] key and cancels the command line |

Depending upon how your system is set up, DOS might prevent you from entering certain control characters using one or more of the methods for creating text files. If that happens, you should try using one of the other methods.

Let's look at how you enter control characters into a text file. Because each method has its own rules for entering control characters, we'll discuss them one at a time.

Entering control characters with ECHO and COPY CON

DOS lets you enter most of the control characters into text files from the command line using either the ECHO or COPY CON command. To enter a control character into a text file, you press [Ctrl] plus its letter or character, or you can hold down the [Alt] key and type its ASCII code number on the numeric keypad. Regardless of which method you use to enter a control character, the character will appear on the command line as a caret followed by the letter. (There are two exceptions: ^T appears as ¶ and ^U appears as § as soon as you press the [Ctrl]-letter or [Alt]-number combination.) For example, you can echo the form feed character, as we did earlier, like this

```
echo <[Alt]12>
```

or

```
echo <[Ctrl]L>
```

It's interesting to note that you can create the ASCII symbol for the ^S control character by pressing [Ctrl]S repeatedly. You press [Ctrl]S three times to create a single !! symbol; if you have Doskey loaded, you must press [Ctrl]S five times to create the symbol.

There are several control characters you can't enter into a text file you create at the command line because DOS interprets them as commands and executes them as soon as you type them. In addition to ^H, ^I, ^J, ^M, ^Z, and ^[, which we mentioned earlier, you can't put the following characters into a text file:

- | | | |
|----|--------------|---|
| ^C | ASCII 3 (␣) | • cancels the current command |
| ^P | ASCII 16 (␣) | • sends whatever you type thereafter to the printer or print buffer until you press [Ctrl]P again |

Depending upon your setup, DOS might not let you enter certain other control characters from the command line. If that's the case, you might try entering them in the Editor instead.

Entering control characters in the Editor

You can create most control characters in the MS-DOS Editor, but there are a few rules to remember. The Editor uses several [Ctrl]-letter combinations

for editing, moving the cursor, and other functions. To enter a control character, you must first press [Ctrl]P to tell the Editor that the next character is a control character. Then you enter the [Ctrl]-letter combination for the character you want to create. For example, you can create the ♥ character by pressing [Ctrl]P and then [Ctrl]C.

As an alternative, you can press [Alt] and type 16 on the numeric keypad followed by the [Alt]-ASCII code of the character you want to create. ([Alt]16 is the ASCII equivalent of the ^P character.) So, you can create the ♥ character by pressing [Alt]16 and then [Alt]3, the heart symbol's ASCII code.

The MS-DOS Editor won't let you add certain control characters using the [Ctrl]P plus [Ctrl]-letter method. For example, you can't create the ^L (form feed) character by pressing [Ctrl]P plus [Ctrl]L. If you press that key combination, the Editor calls up the Repeat Last Find dialog box. Fortunately, you can create ^L by pressing [Ctrl]P or [Alt]16 followed by its ASCII code number, [Alt]12. If the Editor prevents you from adding any character by pressing [Ctrl]P plus [Ctrl]-letter, try the [Alt] method.

We already noted the seven control characters you can't use as graphic characters. Five of them—^G, ^H, ^I, ^Z, and ^[—appear when you enter the control character into a text file but disappear when you type the text file to the screen. Instead, you see the result of that action when you type the text file. You can't even create the other two—^J and ^M—in the MS-DOS Editor. In addition, the Editor doesn't let you create the ^U character.

Of course, your results may differ, so you should try to place any control character in a text file using the [Ctrl]P plus [Ctrl]-letter method or the [Alt]16 plus [Alt]-number method. If you can't add certain characters to a text file using the Editor and you have DOS 5, you can try using Edlin.

Entering characters using DOS 5's Edlin

If you have access to Edlin, you can use it for entering most control characters into a text file. To add a control character to a text file in Edlin's input mode, you press [Ctrl]V followed by the uppercase letter or character that identifies the control character you want to add. (The [Ctrl]V combination tells Edlin to interpret the next character as a control character.) For example, you enter the ^G character by typing [Ctrl]V followed by the capital letter G. Edlin will display your input like this:

```
*i
1: *^VG
```

You must enter control characters in this manner to keep DOS from interpreting the control character as

an editing command. For example, if you try to enter the escape character by pressing [Ctrl][, Edlin cancels the line. If you press [Ctrl]V plus [, Edlin will enter the character into your text file.

It's interesting to note that when you exit Edlin's input mode and list the file, your control characters display without the V prefix. For example, if you entered the ^G character as we did earlier and then use the Edlin List command to list the file, it would display like this:

```
*L
1: ^G
```

As with the other methods, there are certain control characters you can't use as graphic characters because DOS interprets them as editing keys. For example, when you list a file containing these characters, Edlin immediately performs the commands attached to them:

^I	• creates a tab
^J	• creates a line feed
^M	• creates a carriage return

When we exit Edlin and type the text file on our system, several characters seem to disappear. In addition to ^I, ^J, and ^M, you won't see the following characters' ASCII symbols:

^G	• bell
^H	• backspace
^Z	• end-of-file marker
^[• escape character

Notes

We've shown you how to enter control characters and ASCII characters into text files. You might wonder if you can print these characters, too. Well, we can't answer that one—you'll have to check your printer's manuals to determine whether it can print ASCII characters and, if so, how to do it.

We've mentioned that your system might dictate which ASCII and control characters you may enter into a text file. You also might discover that certain characters create unexpected results when you send them to or through a specific device. If this happens, be sure to check the manuals that came with your device for a list of control codes the device recognizes.

Conclusion

If you have difficulty entering ASCII and control characters into text files, you're not alone. DOS offers several methods for creating these characters, and each method carries its own rules. In this article, we showed you how to enter ASCII and control characters using the ECHO and COPY CON commands, the MS-DOS Editor, and DOS 5's Edlin. ♦

Microsoft Technical Support
(206) 454-2030

LETTERS

VERSION
6.x

Disabling DOS 6's [F5] and [F8] keys at startup

I am in charge of 18 computers at a community college campus, and I use DOS's multiple configuration menu. I've had some problems in the past with someone loading viruses on the machines; therefore, I have written a password-protection program to prevent unauthorized access to the DOS prompt.

The problem occurs when the multiple configuration menu appears on the screen—if shows instructions at the bottom telling how to bypass the menuing system and go directly to DOS:

F5=Bypass startup files F8=Confirm each line of
CONFIG.SYS and AUTOEXEC.BAT [N]

Is there a command I can use to rid the screen of these instructions?

*Linda Dickert
Beaufort, South Carolina*

There certainly is. To disable the [F5]/[F8] message at the bottom of your multiple configuration menu, you just add this line to your CONFIG.SYS file:

SWITCHES=/N

You can add the line anywhere in the file, either before or after the [menu] commands. Then reboot and see the results for yourself.

The SWITCHES=/N command disables not only the message at the bottom of the menu but also the functions attached to both the [F5] key and the [F8] key. (You can't disable only one of the keys.) When you press the [F5] key while your system boots up, DOS bypasses your CONFIG.SYS and AUTOEXEC.BAT files. As a result, DOS doesn't load any device drivers, TSRs, or environment variables during startup—you're left with a "plain vanilla" setup. When you press the [F8] key during startup, DOS prompts you to confirm (or deny) each CONFIG.SYS and AUTOEXEC.BAT entry.

You'll want the capability to disable [F5] and [F8] in situations like Ms. Dickert's, where you need to prevent savvy users from bypassing your password-protection scheme simply by pressing a key. You'll also want to disable [F5] and [F8] in situations where new users might be confused by the cryptic message at the bottom of the screen.

You can use the SWITCHES=/N command to disable [F5] and [F8] even if you don't use a configuration menu in DOS 6. Simply add the line anywhere in your CONFIG.SYS file and DOS will disable both keys whenever you boot or reboot your computer.

Regaining control over CONFIG.SYS

So far we've talked about completely disabling your CONFIG.SYS and AUTOEXEC.BAT files by pressing [F5], about pressing [F8] so you can confirm each CONFIG.SYS and AUTOEXEC.BAT entry, and about using SWITCHES=/N to deny user access to both files at startup. Now we'll show you another wrinkle.

DOS 6 lets you set up your CONFIG.SYS file so it will ask you to confirm individual lines of the file at startup. All you have to do is add a question mark (?) after the command and before the equal sign (=) in the entries you want to be able to selectively confirm at startup. For example, the line

INSTALL?=C:\DOS\FASTOPEN.EXE C:=50

in your CONFIG.SYS file lets you choose at bootup whether to run Fastopen. ♦

DOS Software Connection

Are you on the lookout for good DOS shareware, freeware, and public domain software? If so, you may want to subscribe to DOS Software Connection. DOS Software Connection is a service that provides you with a monthly disk loaded with useful DOS utilities, applications, games, and much, much more.

You can purchase a one-year subscription to DOS Software Connection for \$59. Or, if you don't want to subscribe but would like to purchase a single disk, you can do so for only \$7.50. To subscribe to DOS Software Connection or to order a single disk, just call The Cobb Group Customer Relations at (800) 223-8720. Outside the US, please call (502) 491-1900.

